

Timeline reminder system bot and telegram assistant chatbot for a university student and lecturer

Nur Azizul Haqimi¹, Rendra Tri Kusuma²

^{1,2}Diploma 3 Informatics Engineering, Universitas Sebelas Maret, Indonesia

Article Info

Article history:

Received October 1, 2023
Revised October 26, 2023
Accepted October 31, 2023

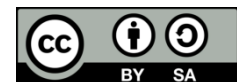
Keywords:

Chatbot
Timeline reminder
System development life cycle
Golang
Codeigniter4

ABSTRACT

Advances in information technology are increasing the use of chatbots for conversation in university academic services. Chatbot conversations are integrated with the activities of lecturers, students and staff to manage academic schedules. The background of this research is due to the difficulties of lecturers in managing activity schedules and answering questions that come from D3 Informatics Engineering study program students effectively and efficiently. This shows that having chatbot reminder and assistant applications will be very useful for study programs. The purpose of this research is to develop a bot and chatbot application that can help lecturers and students at Diploma 3 Informatics Engineering in managing activity schedules and answering questions related to study programs. The research method used is the Waterfall system development method, which is a type of System Development Life Cycle (SDLC). This method follows the sequential stages of system development starting from requirements analysis, system design, implementation, testing and maintenance. Research output is Development of the Timeline Reminder System Bot and Telegram Assistant Chatbot for the UNS Madiun Informatics Engineering D3 Study Program.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Nur Azizul Haqimi Oktavianoro,
Diploma 3 Informatics Engineering,
Universitas Sebelas Maret, Indonesia.
Email: nura.haqimi@gmail.com
<https://doi.org/10.52465/joscecx.v4i4.221>

1. INTRODUCTION

The activities of lecturers, students and educational staff at an institution often have complexities so that establishing a schedule of activities between the three requires mutual agreement. However, manual scheduling often leads to lots of questions, repetitive answers and unstructured conversations. Chatbots are only used on systems that have been determined by the developer, namely Telegram. Chatbots can only process questions asked by detecting the content of words that have been determined by the admin, not processing the meaning of the questions asked.

Reminder is a message that helps someone to remember something. Reminders can be more useful when contextual information is used to present information at the right time and in the right place. Reminders can be used in time management to provide warning alarms in the form of location, time-based notifications

or contextual notes [1]. A timeline reminder is a feature or system that helps remind someone about predetermined events or tasks, usually using a predetermined time as a trigger to provide a reminder. It can be used in various applications, such as task management applications, calendars, and others. The function of the timeline reminder is to help users overcome the problem of forgetting or lack of efficiency in managing daily tasks and schedules[2]. Bot Timeline Reminder application, there are several applications that manage every need, namely synchronous and asynchronous requests. If there is an order that requires instant reply data after the data is sent, it means it will be processed via the Synchronous application. On the other hand, if the data sent requires a pause or schedule to execute it, the data goes to the Asynchronous application, namely through the Publisher application, which will then be processed by the Subscriber. The assistant chatbot runs on the HTTP API of the Telegram application, therefore the chatbot can only be accessed by users who have registered in the Telegram application. Technically, the bot application will authenticate via the installed middleware before continuing to execute data in the main bot application.

Chatbots have been known as an effective technology in providing information services and human-computer interaction. In order to achieve conversational goals effectively, chatbots must proactively provide appropriate responses or feedback to users and guide them to get the information they need by asking specific questions based on request criteria [3]. As is known, chatbots are one of the developments in creating machine-human conversation simulators [4]. Chatbot is also a computer program that is able to continue conversations and answer questions from human users [5]. The context of conversational services is developing into various fields, including services in the business and financial fields[6][7][8], tourism [9][10], communications, messaging services or social media[11], psychology, medicine, and education[12].

The first aim of the research is to develop a Bot Timeline Reminder application that can help lecturers at D3 Informatics Engineering remember and organize schedules of activities related to the study program. The second aim is to develop a Telegram Assistant Chatbot application that can help students find answers about lecture or study program needs and D3 Informatics Engineering in answering messages automatically. Related research is chatbots in education have been used to check students background habits through cognitive or affective empathic communication [13]. Other research related to the field of educational technology is the application of chatbots in courses at a university, with satisfactory results in helping students improve their academic achievement [14][15][16]. Students who have been able to use chatbots in some of their courses have discovered the benefits of using chatbots and requested that these chatbots be used in other aspects of their academic life[17]. The difference from previous research lies in the subject matter, for example, this research discusses academic services, while previous research is related to student learning taxonomies and course materials. Education 4.0 requires us to adapt to advances in information technology, namely chatbots with instant messaging services such as Telegram, WhatsApp. Chatbots can act as learning tutors. Chatbots can be categorized into two, namely task oriented [18] and non task oriented. Non-task oriented chatbots are intended to imitate the characteristics of unstructured conversations between humans and in open domains [19]. Chatbot maintenance by D3 Informatics Engineering can be as easy as managing FAQs (Frequently Asked Questions) and updating spreadsheet type databases. However, university staff need to be convinced of its usefulness, and this can sometimes be an obstacle [20]. Although students do not provide information that can identify them in detail, they provide a variety of information that is considered pseudo-identification and can be used to identify the person in total. In the case study, chatbot development does not collect logs because Google DialogFlow stores logs for 400 days if logging is enabled. And these logs must remain enabled so that the chatbot can continue to learn so it is highly recommended to implement open source tools that allow greater privacy and control by the university over these types of logs [15]. Chatbots make it easier for students to handle university/study program administration issues, such as registering for training, exam schedules, their grades, and those related to their studies so that the pressure on study programs is reduced significantly[21]

2. METHOD

Business processes are a series of activities that form the application work flow and explain how application development will be built. The following is the business process or system flow in the Bot Timeline Reminder application and what interactions are in this application:

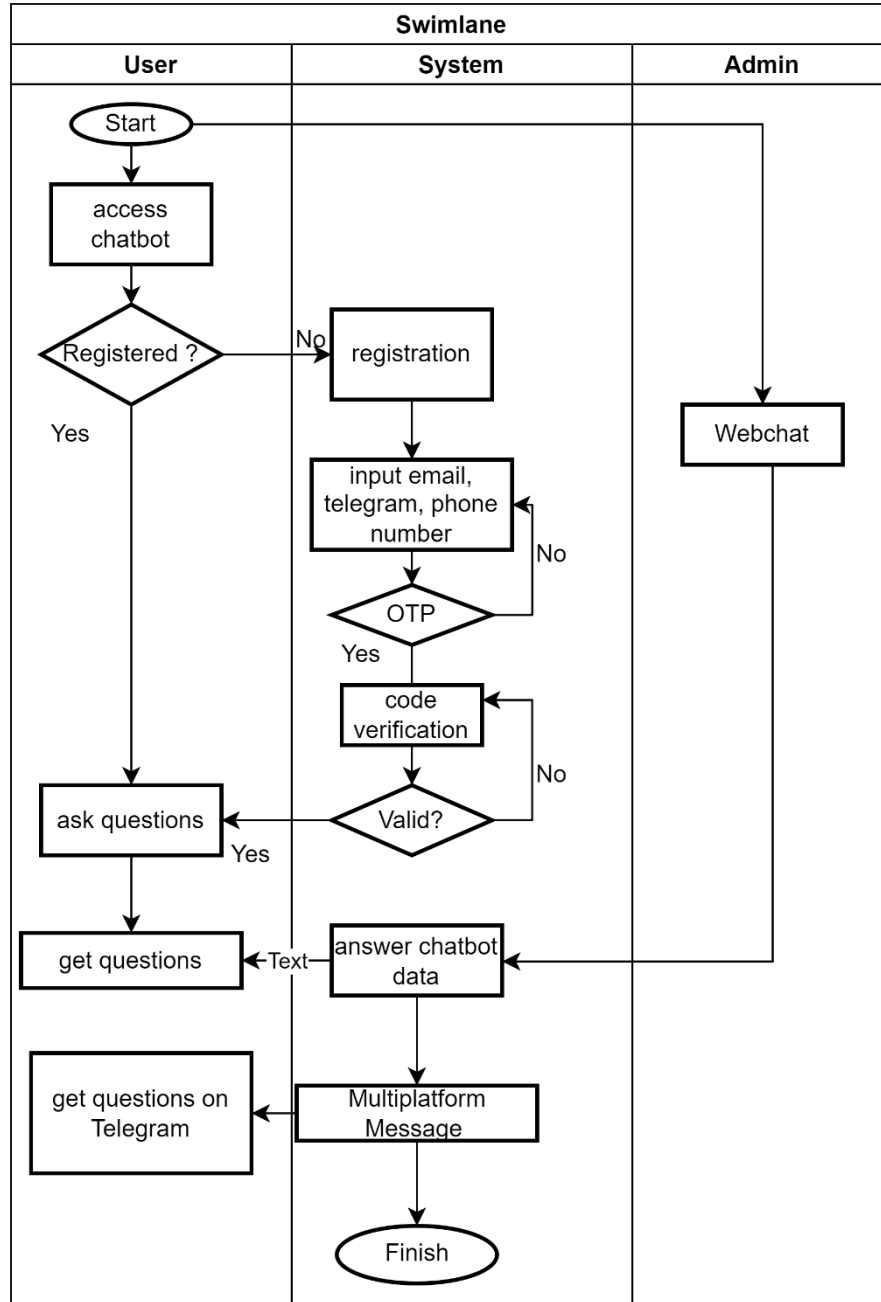


Figure 1. System flowchart

- Admin can log in using the email and password created by the system.
- Admin inputs information on activities to be carried out by the D3 Informatics Engineering study program such as event name, description, PIC and documents as required.
- Admin sets event reminders by entering the time when the activity reminder will be sent to the person in charge of the activity concerned. Admins can also send event reminders manually through the system.
- Admin can send massive messages to all staff or students who have registered in the application.
- Admin can see activities on the application dashboard.
- The system can send multiplatform messages with predefined platforms.

2.1 Assistant Chatbot

The research phases are divided into four sections: Problem, Designing, Data Processing, and Conclusion, in accordance with the order displayed in Figure 1 as follows.

- The chatbot runs within the HTTP API of the Telegram application so the chatbot can be accessed via the Telegram application only.
- The chatbot only responds to users whose data has been registered in the application database.
- Users are provided with a dashboard for registration, verification and management of data registered in the application.
- Users can interact with the chatbot with questions determined by the admin.
- The admin can set the responses/answers that the chatbot will send to users who ask questions in the chatbot, the admin management dashboard becomes one in the admin dashboard of the Bot Timeline Reminder application.

2.2 Application Architecture

Application architecture is a view that describes how an application system is built and how its components interact with each other. Application architecture includes division of tasks, access levels, data flow, and interactions between components. In this point, we will explain the architectural design of the Timeline Reminder Bot application and also the Chatbot Assistant. If you look at it as a standalone application, the application architecture will be seen in Figure 1. The architecture is as follows.

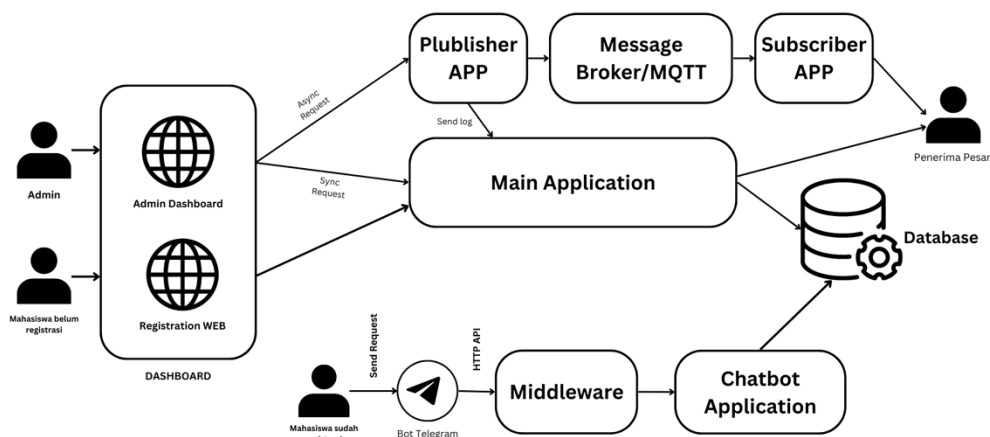


Figure 1. Timeline reminder and chatbot architecture

The chatbot algorithm used to answer questions using tags has the goal of matching user questions with relevant answers. The flow of this algorithm can be illustrated with a flow image as in the image provided. First of all, the chatbot algorithm will receive questions from the user. This question will be processed to extract the tags or keywords contained in it. These tags will be used to search for relevant answers in the database provided by the admin. Every answer provided by the admin will be given relevant tags. For example, if the answer contains a general lecture schedule, then the relevant tags could be "public lecture schedule", "public lectures", and so on. These answers will be associated with their tags in the chatbot database.

When a question is received, the algorithm will analyze the question and extract the tags contained in it. For example, if the question contains the words "public lecture" and "schedule", then the relevant tags are "public lecture" and "schedule". Next, the algorithm will match the tags found in the question with the tags associated with the answers in the database. If there is more than one answer that has a tag that is relevant to the question, the algorithm will use the tag that appears most frequently in the question (frequently tag) as a reference for selecting the most relevant answer. Once the most relevant answer has been selected, the chatbot will provide the user with an answer based on that answer.

By using this algorithm, chatbots can provide more accurate and relevant responses to user questions. By connecting question tags with answer tags, the system can match questions with relevant answers based on the keywords contained in them. This allows the chatbot to provide answers that suit the user's needs more effectively.

System implementation is the stage for implementing the design that has been carried out on the system so that it is ready to operate. In this project, the total required is approximately 4 applications that run independently which will be explained in the points below. In making all these applications there are 2 communication methods for processing the data, namely Synchronous and Asynchronous (Event-Driven).

2.3 Synchronous Implementation

In this project, the implementation of the synchronous method is used for processes that often occur in applications in general, namely the application will process the data request desired by the user after the request command is issued, for example when we make a request to an application, for example a page that displays data, then when we open the page or click a button which will trigger the appearance of the data, meaning the process runs synchronously, meaning there is a user who requests the appearance of the data, the system will provide the data according to the request and it will be processed immediately after the request and will be sent after the data retrieval process is successful. , while from the user's side he waits until the request data has finished processing before the data will appear.

2.4 Asynchronous (Event-Driven) Implementation

In the previous point, we discussed the implementation of the synchronous method used in this project. However, apart from synchronous methods, this project also applies asynchronous or event-driven methods to overcome several cases that require a different approach.

In an asynchronous implementation, the system will receive and process requests from users without having to wait for the results to be returned directly. This means the system will queue the request or set an event listener to capture the request, and then move on to handle the next request without having to wait for the results.

One example of using asynchronous methods is when an application must perform a task that requires a long time, such as contacting an external source that requires uncertain communication times, heavy processing, or other network operations. In such cases, using a synchronous method will make users have to wait a long time without any clear feedback.

In an asynchronous implementation, the system will receive a request from the user and start the processing task without waiting for the result. After the task has finished processing, the system will send feedback or a signal to the user that the process has been completed. This allows users to continue interacting with the application without having to wait for a long task to complete first.

In this project, asynchronous or event-driven methods are implemented using technologies such as message queues or pub/sub (publish-subscribe) systems. The user will make a request, and the system will put the request into a message queue or pub/sub system. After the task has finished processing, the system will send a message or event to the user to notify that the job is complete.

Asynchronous implementations provide greater flexibility in handling requests that take a long time, optimize resource usage, and improve application responsiveness. However, it should be noted that asynchronous use also requires good management to avoid problems such as message queues that are too long or unsuccessful message delivery.

In this project, asynchronous (event-driven) methods are used to handle certain cases where a fast response or handling of tasks that require a long time is required or a long process. The implementation involves the use of technology such as a message queue/message broker called a pub/sub system to ensure reliable message delivery and proper settings for handling asynchronous requests.

2.5 Databases

In this project, the MySQL database is used as a database management system (DBMS) to store and manage the data required by the application. MySQL is a popular DBMS and is often used in web application development. Implementing a MySQL database involves several important steps, including:

- a) MySQL Installation: The first step is to install MySQL on the server or environment that will be used. After the installation is complete, the MySQL database server will be running and ready to use.
- b) Database Schema Design: The next step is to design the database schema that will be used in this project. The database schema includes the tables that will be used, the columns in each table, as well as the

relationships and rules that govern between these tables. Database schema can be represented in the form of an ER (Entity-Relationship) diagram or in other formats.

- c) **Table Creation:** After the database schema is determined, the next step is to create tables in the MySQL database according to the schema that has been designed. Each table will have columns that correspond to the attributes needed to store data.
- d) **Definition of Relationships and Keys:** Next, you need to define the relationships between the tables in the database. This is done using the concept of foreign keys, where columns in one table refer to other columns in a different table. This definition of relationships and keys helps maintain data integrity and perform join operations between tables.
- e) **CRUD Operations:** Once the tables are created, you can perform CRUD (Create, Read, Update, Delete) operations on the data in the MySQL database. By using a query language such as SQL (Structured Query Language), you can perform operations - insert new data, select to read data, update to change data, and delete to delete data.
- f) **Database Connection:** Applications that use a MySQL database need to connect to the database server to access and manipulate data. This involves setting up a connection to the database using a MySQL connector that corresponds to the programming language used in the application.

With a proper MySQL database implementation, you can store and manage data efficiently in this project. MySQL provides various features and capabilities that enable reliable data processing and scale to suit application needs.

2.6 Telegram chatbots

Telegram is a cloud-based instant messaging application that is free and available on iOS and Android mobile devices. Telegram is open-source with free chatbot hosting facilities and cross-platform [22]. Implementation of the Telegram chatbot begins with registering a bot account with the Telegram service provider. Permission to create bots can be registered via Bot Father. Bot Father is a service from Telegram for providing access rights and creating bot addresses. If the Telegram bot has been successfully created, a token to access and manage the chatbot will be available via the endpoint. Available bots can be searched for using the search bar on the main Telegram page by typing in the keywords for the bot's name.

So that the response given by the chatbot can be regulated, it can be controlled directly through the Golang program using the endpoint token provided by Telegram. This token is secret because with this bot token we can fully manage the registered chatbot. An additional library is needed, namely telegram-bot-api so that our Golang program can connect to registered bots. In the program we need to enter information in the form of the telegram bot token. From this token and connected to the bot's endpoint, we can capture every chat update given to the chatbot. From these chat updates, we can capture the contents of the chat, the sender and other information about the sender's account. This information can be processed according to subsequent needs. In this case the chatbot should be able to respond to questions asked via chat, provided that the questions or in this case we call tags have been registered in the database, so that the chatbot can respond as desired, it needs to create a function to match the questions asked with the database of answers provided. has been provided.

In this case, the chatbot will enter the registration data into the user database with pre-arranged message conditions. The registration requirements and what commands can trigger the bot to enter user data will be discussed in the next point regarding the restful api application endpoint. An example of an account registration message is as follows:



Figure 3. Telegram chatbot display

For user registration to be verified in the previous point, must be able to get the verification message which has been prepared through the specified portal. That way, the user is registered with the chatbot and can interact with roles that are automatically detected by the system as Students or Staff. So some branching of functions is needed so that messages sent via chatbot can be responded to according to the flow, namely when the user has not been verified, the user who sent the account registration command and the user who has been verified.

2.7 System Testing

System testing is an important stage in application development to ensure that the system being built functions well, complies with predetermined requirements, and can face various situations that may occur.

Message Broker Testing

In message broker testing, the main focus is to ensure that the communication system between components that use the message broker runs well. Some of the aspects tested in this test include:

1) Connection Testing

In connection testing, the system address of the message broker is needed. To use the RabbitMQ message broker, this address is usually on port 5672 by default, while for the admin page it is on port 15672. For testing this time we will focus on port 5672 because that is the port for creating events in RabbitMQ, testing is carried out by creating a unit test for connection testing and the test results that successfully connect to RabbitMQ look like the following:

2) Testing New Channel Creation

This test is to test whether we can create a new channel in RabbitMQ or not because this channel is very important for separation between queues when the Pub/Sub system is created. If these two tests are successful, the message broker can be used to create a new queue for the Event-Driven system.

3) Restful API Testing

Restful API testing is used to test whether the endpoint that has been created runs according to the desired flow or not. Endpoint testing can be done by creating an HTTP file and writing the test information manually or with a third-party application for the test. In this test I used a third-party application for testing, namely Postman with test results such as Chatbot Register Endpoint, Send Email Endpoint, Multiplatform Notification Endpoint and Scheduled Multiplatform Notification Endpoint. All endpoint tests above for each endpoint are functioning well so they can be integrated with the client application that will use the endpoint.

Website Testing

Website testing is carried out by testing the functional features created using test scenarios created. This test includes the web for Chatbot and also Timeline Reminder. The chatbot dashboard is a website for users, both students and lecturers, to register into the chatbot application and also to view the data registered in the chatbot.

Telegram Chatbot Testing

This test was carried out to ensure that the Telegram chatbot was connected to the system and could also answer chats from users whose answers had been set on the chatbot management website. There are several test cases carried out, namely tests when users have not yet registered, users who request registration/register an account and users who ask questions.

User Asks Questions

Testing is carried out for several cases, because for the process of answering questions from users using frequently tags, it is necessary to try when the question asked can contain more than one answer available in the database.

1) Preparation

Make 2 answers, the first answer is "final assignment hearing schedule" which has the tags "trial" and "trial schedule", the second answer is "This is the final assignment hearing schedule for July" which has the tags "trial", "trial schedule", "July session" and "July session schedule" and attached PDF documents. It can be seen from the two answers that they contain similar tags, namely the words trial and trial schedule.

2) Test send question 1

The first experiment sends a message containing 1 tag for each answer.



Figure 4. Testing negative case

It can be seen from the image above that when there is a question "trial" which means the word contains the tag "trial" in both answers, and there are no other words following it, then the question is considered ambiguous by the chatbot because it can have 2 possible answers, therefore it will be answered incomplete messages by chatbot.

3) Test send question 2

The second attempt sent a message with the message "Give me the final assignment trial schedule" from this question there were 2 tags that were the same in the 2 answers, namely "trial" and "trial schedule", following were the test results:



Figure 4. Testing positive case

3. RESULTS AND DISCUSSIONS

System testing using three tests, namely Message Broker, restful API, website. Message broker testing, the main focus is to ensure that the communication system between components that use the message broker runs well. Some of the aspects tested in this test include connection and creation of new channels.

Restful API testing is used to test whether the endpoint that has been created runs according to the desired flow or not. Endpoint testing can be done by creating an HTTP file and writing down the test manual information or with a third-party application for the test. In this test I used a third-party application for testing, namely Postman with test results as follows: Endpoint Register Chatbot, Endpoint Send Email. Website testing is carried out by testing the functional features created using test scenarios created. This test includes the web for Chatbot and also Timeline Reminder.

4. CONCLUSION

From the three tests, the chatbot and timeline maker can run well using FAQs for study program activities. Chatbots and timeline reminders for study programs are very helpful in compiling documents used for accreditation. does not rule out the possibility of this system developing with the adoption of the latest information technology methods such as Machine Learning.

The method we use is the Waterfall system development method which will help us ensure that the applications we produce comply with the objectives and specifications set. In the future, we will develop this chatbot using Machine Learning, especially Tag Similarity. We are confident that the results of this research will provide benefits to the D3 Informatics Engineering study program and also to the author.

REFERENCES

- [1] N. Ramsari And A. Rifaldi, "Rancang Bangun Aplikasi Penjadwalan Kegiatan Akademik Disertai Sistem Reminder Berbasis Responsive Web Design," *J. Teknol. Inf. Dan Komun.*, Vol. 9, No. 1, 2018.
- [2] I. Handayani, Q. Aini, And Y. Oktavyanti, "Penggunaan Rinfocal Sebagai Aplikasi Pengingat (Reminder) Kegiatan Akademik Perkembangan Teknologi Informasi Memberikan Pengaruh Yang Sangat Besar Di Dalam Kehidupan Sehari-Hari Di Sekitar Dinding Dan Kalender Meja , Dan Ada Pula Terdapat Di Ponsel . Pe," Vol. 9, No. 1, Pp. 13–26, 2015.
- [3] A. P. Chaves And M. A. Gerosa, "How Should My Chatbot Interact? A Survey On Social Characteristics In Human–Chatbot Interaction Design.Tle," *Int. J. Human–Computer Interact.*, Vol. 37, No. 8, Pp. 729–758, 2021.

- [4] E. Nila And I. Afrianto, "Rancang Bangun Aplikasi Chatbot Informasi Objek Wisata Kota Bandung Dengan Pendekatan Natural Language Processing," *J. Ilm. Komput. Dan Inform.*, Vol. 4, P. 6, 2015.
- [5] M. Sharma, S. Verma, And L. Sahni, "Comparative Analysis Of Chatbots," *Ssrn Electron. J.*, 2020, Doi: 10.2139/SSrn.3563674.
- [6] S. Hwang And J. Kim, "Toward A Chatbot For Financial Sustainability," *Sustainability*, Vol. 13, No. 6, 2021, Doi: 10.3390/Su13063173.
- [7] T. Um, T. Kim, And N. Chung, "How Does An Intelligence Chatbot Affect Customers Compared With Self-Service Technology For Sustainable Services?," *Sustainability*, Vol. 12, No. 12, 2020, Doi: 10.3390/Su12125119.
- [8] S. Khan And M. R. Rabbani, "Chatbot As Islamic Finance Expert (Caife): When Finance Meets Artificial Intelligence," In *Proceedings Of The 2020 4th International Symposium On Computer Science And Intelligent Control*, In Iscsic 2020. New York, Ny, Usa: Association For Computing Machinery, 2021. Doi: 10.1145/3440084.3441213.
- [9] M. Carisi, A. Albarelli, And F. L. Luccio, "Design And Implementation Of An Airport Chatbot," In *Proceedings Of The 5th Eai International Conference On Smart Objects And Technologies For Social Good*, In Goodtechs '19. New York, Ny, Usa: Association For Computing Machinery, 2019, Pp. 49–54. Doi: 10.1145/3342428.3342664.
- [10] F. P. Putri, H. Meidia, And D. Gunawan, "Designing Intelligent Personalized Chatbot For Hotel Services," In *Proceedings Of The 2019 2nd International Conference On Algorithms, Computing And Artificial Intelligence*, In Acai '19. New York, Ny, Usa: Association For Computing Machinery, 2020, Pp. 468–472. Doi: 10.1145/3377713.3377791.
- [11] C. Frommert, A. Häfner, J. Friedrich, And C. Zinke, "Using Chatbots To Assist Communication In Collaborative Networks," In *Collaborative Networks Of Cognitive Systems: 19th Ifip Wg 5.5 Working Conference On Virtual Enterprises, Pro-Ve 2018, Cardiff, Uk, September 17-19, 2018, Proceedings 19, 2018*, Pp. 257–265.
- [12] A.-A. Georgescu, "Chatbots For Education--Trends, Benefits And Challenges," In *Conference Proceedings Of\Guillemotright\ Elearning And Software For Education {Guillemotleft}(Else)*, 2018, Pp. 195–200.
- [13] C. Zhai And S. Wibowo, "A Systematic Review On Cross-Culture, Humor And Empathy Dimensions In Conversational Chatbots: The Case Of Second Language Acquisition," *Heliyon*, Vol. 8, No. 12, P. E12056, 2022, Doi: 10.1016/J.Heliyon.2022.E12056.
- [14] S. H. Mad Daud, "E-Java Chatbot For Learning Programming Language: A Post-Pandemic Alternative Virtual Tutor," *Int. J. Emerg. Trends Eng. Res.*, Vol. 8, No. 7, Pp. 3290–3298, 2020, Doi: 10.30534/Ijeter/2020/67872020.
- [15] A. Balderas, R. F. Garcia-Mena, M. Huerta, N. Mora, And J. M. Dodero, "Chatbot For Communicating With University Students In Emergency Situation," *Heliyon*, Vol. 9, No. 9, P. E19517, 2023, Doi: 10.1016/J.Heliyon.2023.E19517.
- [16] D. Lee And S. Yeo, "Developing An Ai-Based Chatbot For Practicing Responsive Teaching In Mathematics," *Comput. Educ.*, Vol. 191, P. 104646, Dec. 2022, Doi: 10.1016/J.Compedu.2022.104646.
- [17] M. Ismail And A. Ade-Ibijola, "Lecturer's Apprentice: A Chatbot For Assisting Novice Programmers," *2019 Int. Multidiscip. Inf. Technol. Eng. Conf. (Imitec), Vanderbijlpark, South Africa*, 2019, Doi: 10.1109/Imitec45504.2019.9015857.
- [18] K. Yoshino, "Dialogue System Technology Challenge: From Dialogue State Tracking To Dialogue System Technology In Real World," *J. Acoust. Soc. Japan*, Vol. 79, No. 9, Pp. 477–483, 2023, Doi: 10.20697/Jasj.79.9_477.
- [19] A. Meloni, S. Angioni, A. Salatino, F. Osborne, D. Reforgiato Recupero, And E. Motta, "Integrating Conversational Agents And Knowledge Graphs Within The Scholarly Domain," *Ieee Access*, Vol. 11, No. March, Pp. 22468–22489, 2023, Doi: 10.1109/Access.2023.3253388.
- [20] C. I. Maican, A. M. Cazan, R. C. Lixandriou, And L. Dovleac, "A Study On Academic Staff Personality And Technology Acceptance: The Case Of Communication And Collaboration Applications," *Comput. Educ.*, Vol. 128, No. March 2018, Pp. 113–131, 2019, Doi: 10.1016/J.Compedu.2018.09.010.
- [21] E. Adamopoulou And L. Moussiades, "Chatbots: History, Technology, And Applications," *Mach. Learn. With Appl.*, Vol. 2, No. July, P. 100006, 2020, Doi: 10.1016/J.Mlwa.2020.100006.
- [22] S. Chng, J. Plananska, And L. Cheah, "Comparison Of Travel Attitude Study Methods Using Online Tools: The Case Of Understanding Public Acceptance Of Autonomous Vehicles," *Transp. Res. Interdiscip. Perspect.*, Vol. 20, P. 100847, Jul. 2023, Doi: 10.1016/J.Trip.2023.100847.